Large Language Models Are Zero-Shot Fuzzers: Fuzzing Deep-Learning Libraries via Large Language Models

Year: 2022 | Citations: 276 | Authors: Yinlin Deng, Chun Xia, Haoran Peng, Chenyuan Yang, Lingming Zhang

---

Abstract

Deep Learning (DL) systems have received exponential growth in popularity and have become ubiquitous in our everyday life. Such systems are built on top of popular DL libraries, e.g., TensorFlow and PyTorch which provide APIs as building blocks for DL systems. Detecting bugs in these DL libraries is critical for almost all downstream DL systems in ensuring effectiveness/safety for end users. Meanwhile, traditional fuzzing techniques can be hardly effective for such a challenging domain since the input DL programs need to satisfy both the input language (e.g., Python) syntax/semantics and the DL API input/shape constraints for tensor computations. To address these limitations, we propose TitanFuzz – the first approach to directly leveraging Large Language Models (LLMs) to generate input programs for fuzzing DL libraries. LLMs are titanic models trained on billions of code snippets and can autoregressively generate human-like code snippets. Our key insight is that modern LLMs can also include numerous code snippets invoking DL library APIs in their training corpora, and thus can implicitly learn both language syntax/semantics and intricate DL API constraints for valid DL program generation. More specifically, we use both generative and infilling LLMs (e.g., Codex/InCoder) to generate and mutate valid/diverse input DL programs for fuzzing. Our experimental results demonstrate that TitanFuzz can achieve 30.38%/50.84% higher code coverage than state-of-the-art fuzzers on TensorFlow/PyTorch. Furthermore, TitanFuzz is able to detect 65 bugs, with 44 already confirmed as previously unknown bugs. This paper demonstrates that modern titanic LLMs can be leveraged to directly perform both generation-based and mutation-based fuzzing studied for decades, while being fully automated, generalizable, and applicable to domains challenging for traditional approaches (such as DL systems). We hope TitanFuzz can stimulate more work in this promising direction of LLMs for fuzzing.